

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

EV444355343

**System and Method for Synchronizing Objects
Between Two Devices**

Inventor(s):
Charles Wu

ATTORNEY'S DOCKET NO. MS1-347USC1

RELATED APPLICATIONS

This application is a divisional application of co-pending U.S. Patent Application No. 09/326,163, filed June 4, 1999, entitled "System and Method for Synchronizing Objects Between Two Devices," to Charles Wu, which claims priority to U.S. Provisional Application No. 60/103,859, filed October 12, 1998, entitled "Flexible Grouping of Objects During Synchronization", to Charles Wu.

TECHNICAL FIELD

This invention relates to synchronizing one or more objects between two computing devices. More particularly, the invention relates to selectively synchronizing an object based on the accessibility of the storage volume that contains the object.

BACKGROUND OF THE INVENTION

Laptop, handheld, and other portable computers or computing devices have increased in popularity as the devices have become smaller in size and less expensive. Additionally, improved operating speed and processing power of portable computers has increased their popularity. Many portable computers are capable of storing multiple application programs, such as address books, games, calculators, and the like. The application programs can be permanently installed in the portable computer during manufacture (e.g., on read-only memory (ROM)). Alternatively, one or more application programs may be installed by the user after purchasing the portable computer.

Many of these small computers have limited physical resources. For example, both primary and secondary memory are typically quite limited in

1 comparison to desktop computers. In addition, small computers and other
2 information processing devices often do not accommodate any form of removable
3 mass storage such as floppy disks or optical disks. To make up for this deficiency,
4 such computers are often capable of utilizing the resources of desktop computers
5 or other base computers.

6 Initially, a base computer (such as a desktop computer) installs application
7 programs on a smaller, more resource-limited portable computer, such as a laptop,
8 handheld, or palmtop computer. Such application programs are typically
9 distributed from their manufacturers on some type of non-volatile storage medium
10 such as a floppy disk or a CD-ROM. Since the portable computer typically has no
11 hardware to read such a storage medium, the portable computer is instead
12 connected to communicate with the base computer, typically through a serial link.
13 The base computer reads the application program from the non-volatile storage
14 medium and downloads the program to the portable computer.

15 Portable computers that can receive application programs downloaded from
16 a desktop computer are versatile and allow application programs to be replaced or
17 upgraded easily. Typically, an installation application is run on the desktop
18 computer that allows the user to select one or more application programs for
19 downloading into the portable computer. After selecting the appropriate
20 application programs, the installation application downloads the application
21 programs to the portable computer.

22 The invention described herein relates to the synchronization of objects,
23 such as databases, stored in portable computers with corresponding objects stored
24 in a base computer. Some portable computers contain a built-in main memory as
25 well as one or more slots or connectors to receive optional removable memory

1 cards. Such memory cards allow a user to increase the memory resources of a
2 portable computer. The additional memory resources can be used for storing one
3 or more objects, storing additional application programs, or executing additional
4 application programs simultaneously. The memory cards are removable from the
5 portable computer, such that the objects or applications stored on the cards will
6 become inaccessible if the card is removed or disconnected from the portable
7 computer. Inaccessible objects cannot be synchronized with the corresponding
8 objects on the base computer because the objects cannot be retrieved unless the
9 memory card is coupled to the portable computer.

10 Typically, when a portable computer is synchronized with a base computer,
11 objects that have been modified since the last synchronization process are
12 synchronized such that the portable computer and the base computer contain
13 identical objects. Further, during each synchronization process, if an object has
14 been deleted on the portable computer or the base computer since the last
15 synchronization process, then the corresponding object on the other system is also
16 deleted. Thus, if a memory card containing a previously synchronized object is
17 removed from the portable computer, then a synchronization process will delete
18 the previously synchronized object from the base computer. Typically, the user of
19 the system did not intend for the objects on the memory card to be deleted from
20 the base computer during a synchronization process. For example, the user may
21 have temporarily removed the memory card to allow the insertion of a different
22 memory card containing different objects or application programs. In this
23 example, the user has not deleted the object from the memory card. The object
24 remains stored on the memory card, but the memory card has been temporarily
25 removed from the portable computer.

1 Although the memory card containing a particular object was removed
2 from the portable computer, the user may desire to continue accessing the object
3 stored on the memory card using the base computer. However, if the object is
4 deleted from the base computer during a synchronization process, the user must
5 re-insert the memory card in the portable computer and complete a
6 synchronization process to allow access to the object using the base computer. If
7 the memory card containing the object is then removed from the portable
8 computer, the next synchronization process will again delete the object from the
9 base computer.

10 Therefore, it is desirable to provide a mechanism that prevents the
11 synchronization of particular objects when one instance of the object is stored on a
12 memory card or other storage device that has become inaccessible to the base
13 computer or the portable computer.

14 15 **SUMMARY OF THE INVENTION**

16 The invention described herein selectively synchronizes objects between
17 two devices. The synchronization is performed such that an object stored on a
18 storage device (such as a memory card) that has become inaccessible to a portable
19 computer is not synchronized with a base computer, thereby preventing the
20 deletion of the object from the base computer. Although the object on the
21 inaccessible storage device is not synchronized, the base computer continues to
22 monitor and record changes made to the corresponding object stored on the base
23 computer. After the storage device becomes accessible (e.g., is re-inserted into the
24 portable computer), a synchronization process is performed such that the two
25 instances of the object are again synchronized. This configuration allows the user

1 to continue accessing an object through the base computer, even when the storage
2 device on which the object is stored is no longer accessible to the portable
3 computer. Thus, the user of the portable computer can temporarily remove storage
4 cards from the portable computer without concern that objects stored on the
5 removed card will be deleted from the base computer.

6 In a particular implementation of the invention, objects are synchronized
7 between a base computer and a portable computer. The portable computer is
8 capable of communicating with a storage volume that can become inaccessible to
9 the portable computer. Storage volumes currently accessible to the portable
10 computer are identified, and only objects contained in those identified storage
11 volumes are synchronized with the base computer.

12 In another implementation of the invention, the synchronization process
13 ignores objects stored on storage volumes that are not currently accessible to the
14 portable computer.

15 Using another aspect of the invention, the base computer continues to
16 monitor and record changes to objects stored on storage volumes that are
17 inaccessible to the portable computer. These changes are synchronized with the
18 portable computer when the previously inaccessible storage volume containing the
19 object becomes accessible.

20 21 **BRIEF DESCRIPTION OF THE DRAWINGS**

22 Fig. 1 illustrates an exemplary portable computer and an exemplary base
23 computer in accordance with the invention.

24 Fig. 2 is a block diagram showing pertinent components of a base computer
25 in accordance with the invention.

1 Fig. 3 illustrates an embodiment of a portable computer in accordance with
2 the present invention.

3 Fig. 4 is a block diagram illustrating pertinent components of a portable
4 computer in accordance with the invention.

5 Fig. 5 is an architectural diagram of a system in accordance with the
6 invention for synchronizing objects between a portable computer and a desktop
7 computer.

8 Fig. 6 illustrates multiple volumes currently stored on a portable computer
9 and a desktop computer.

10 Fig. 7 is a flow diagram illustrating an exemplary procedure for
11 synchronizing objects between a portable computer and a desktop computer.

12 13 **DETAILED DESCRIPTION**

14 Fig. 1 illustrates an exemplary portable computer 100 and an exemplary
15 desktop computer 102 in accordance with the invention. Desktop computer 102 is
16 also referred to herein as a “base computer.” Portable computer 100 can be any
17 type of laptop, palmtop, handheld, or other computing device capable of receiving
18 application programs from a base computer such as desktop computer 102.

19 Portable computer 100 includes a portable synchronization manager 104,
20 which is responsible for coordinating synchronization of objects stored on the
21 portable computer with corresponding objects on base computer 102. An object
22 can be a database or any other data structure capable of being synchronized
23 between two computing devices and/or storage devices. Each object contains
24 multiple data items (also referred to as “data entries” or “records”). The term
25 “synchronization” refers to a process in which changes to one database are

1 automatically reflected in one or more separately stored copies of the database. In
2 the described embodiment, synchronization involves two copies of a database,
3 each containing multiple corresponding entries, items, or records. Changes might
4 be made to an entry in one of the database copies. During synchronization, those
5 changes are implemented on the corresponding entry residing on the other
6 database copy. If the same entry has been changed on both databases, the user is
7 prompted to resolve the conflict by selecting one of the different versions of the
8 entry to discard. When a new entry is created in one of the databases, it is
9 duplicated on the other database during synchronization. When an entry is deleted
10 from one database, it is deleted from the other database during synchronization.

11 The base computer and portable computer are each capable of executing
12 multiple different application programs. Different object types can be associated
13 with each application. For example, a personal contact manager application
14 utilizes an associated object which is a database containing contact information
15 accessed by the contact manager application. Depending on the number of contact
16 entries, the contact manager application may create multiple objects (i.e.,
17 databases) – one for each category of contacts. In a particular example, the
18 contact manager application creates two objects, one for storing personal contact
19 information and another for storing work-related contact information.

20 Portable computer 100 includes a limited amount of built-in memory 110 as
21 well as one or more removable memory cards 112. Removable memory cards 112
22 may also be referred to as “storage cards” or “memory expansion units.” A
23 portion of built-in memory 110 is addressable memory for program execution, and
24 the remaining portion is used to simulate secondary disk storage. The removable
25 memory cards 112 may contain permanently installed applications, such as

1 applications stored in a read-only memory (ROM), not shown. Additionally, a
2 removable memory card 112 may contain non-volatile memory for storing objects
3 (such as databases) or downloaded application programs, thereby supplementing
4 built-in memory 110. Memory cards 112 allow the user of portable computer 100
5 to customize the device by adding application programs or adding memory for
6 storing additional objects and downloading additional application programs.

7 Portable computer 100 typically contains one or more applications 108.
8 Applications 108 may include word processing applications, spreadsheet
9 applications, contact manager applications, and game applications. Although
10 shown as a separate block in Fig. 1, each application 108 is stored in built-in
11 memory 110 or in a removable memory card 112. For each application 108
12 executing on portable computer 100, an application synchronization module 106 is
13 provided. The application synchronization module 106 is familiar with the objects
14 used by the associated application 108. This object knowledge is used by
15 application synchronization module 106 and communicated to portable
16 synchronization manager 104 during the synchronization process.

17 Each storage device in portable computer 100 is divided into one or more
18 storage volumes. A storage volume contains one or more objects capable of being
19 synchronized with corresponding objects in the base computer 102. In one
20 embodiment, the operating system of the portable computer 100 or the base
21 computer 102 is responsible for creating and defining storage volumes. Input
22 from the user of the portable computer or base computer can influence the creation
23 and definition of storage volumes. In other embodiments, the application
24 synchronization module 106 is responsible for creating and defining storage
25 volumes based on its knowledge of the associated application 108.

1 In an exemplary portable computer 100, a removable memory card 112 is
2 represented as a single storage volume and contains one object – a database used
3 by a contact manager application. Another removable memory card 112 in the
4 portable computer 100 is also represented as a single storage volume, but contains
5 multiple objects, such as a separate object for each stock portfolio database used
6 by a portfolio tracking application. The built-in memory 110 of the portable
7 computer 100 is divided into three storage volumes, in which each storage volume
8 is used by a different type of application (e.g., an appointment application, a task
9 list application, and a notepad application).

10 Each storage volume is assigned a globally unique identifier (GUID) – also
11 referred to as a universally unique identifier (UUID). The assignment of a GUID
12 is necessary to properly track all storage volumes that may become accessible to
13 the portable computer. In one implementation of the invention, the GUID is a 16
14 byte identifier generated by the operating system upon creation of the storage
15 volume. In addition to the GUID, each storage volume typically has a name (such
16 as a file name) that is not necessarily unique. Thus, two different memory cards
17 may be named “stock_portfolios”, but the two memory cards will have different
18 identifiers. In addition to volume identifiers, each object has an associated object
19 identifier. Each object stored on the portable computer 100 has an associated
20 identifier and each object stored on the base computer 102 has an associated
21 identifier. Typically, the two identifiers are not identical, thereby requiring a
22 mapping table or similar mechanism for correlating the two object identifiers.
23 Although the volume identifiers are unique, the individual objects stored on the
24 storage volumes do not require unique identifiers.
25

1 Portable computer 100 is designed to take advantage of a base computer's
2 hardware resources. Particularly, portable computer 100 is designed so that
3 application programs and other data can be read from a distribution medium by
4 base computer 102, and then downloaded to portable computer 100. Portable
5 computer 100 is thus referred to as a peripheral computer or an auxiliary computer,
6 in that it is controlled during this process by base computer 102.

7 To allow communications between base computer 102 and portable
8 computer 100, the two computers are coupled to one another through a
9 communication link 114. Typically, communication link 114 is a temporary
10 bidirectional communication link established to exchange data between portable
11 computer 100 and base computer 102. Communication link 114 is used, for
12 example, to synchronize objects between base computer 102 and portable
13 computer 100. Communication link 114 can also be used to download
14 applications and other data from base computer 102 to portable computer 100. In
15 a particular embodiment, communication link 114 is a serial communication link.
16 However, communication link 114 can utilize any type of communication medium
17 and any type of communication protocol to exchange data between portable
18 computer 100 and base computer 102.

19 Base computer 102 in the described embodiment is a conventional personal
20 desktop computer. However, other types of computers might be used in this role.
21 Base computer 102 includes a desktop synchronization manager 116, which
22 operates in combination with portable synchronization manager 104 in portable
23 computer 100 to coordinate the synchronization of objects between base computer
24 102 and portable computer 100. As discussed in greater detail below, desktop
25 synchronization manager module 116 also maintains the status of each storage

1 volume on the portable computer 100. If a particular storage volume in portable
2 computer 100 is not accessible, then desktop synchronization manager 116 does
3 not attempt to synchronize objects stored in the inaccessible volume.

4 Base computer 102 typically contains multiple applications 120.
5 Applications 120 may include applications similar to applications 108 stored on
6 portable computer 100 as well as other applications not associated with the
7 synchronization process. For each application on base computer 102, an
8 application synchronization module 118 is provided. The application
9 synchronization module 118 is coupled to desktop synchronization manager 116
10 and its associated application 120. The application synchronization module 118 is
11 familiar with the objects used by the associated application 120. This object
12 knowledge is used by application synchronization module 118 and communicated
13 to desktop synchronization manager 116 during the synchronization process.
14 Specifically, desktop synchronization manager 118 determines which objects are
15 stored on storage volumes that are accessible to the portable computer 100 and
16 synchronizes only those accessible objects. Base computer 102 also comprises a
17 desktop data store 122 coupled to the desktop synchronization manager 116. Data
18 store 122 stores information necessary to perform the synchronization process,
19 such as the status (e.g., accessible or inaccessible) of each storage volume used
20 with portable computer 100.

21 Although an object stored on portable computer 100 can be synchronized
22 with a corresponding object on base computer 102, the two objects do not
23 necessarily share the same data structure or data storage format. For example, an
24 object stored on the portable computer 100 is created using a contact management
25 program proprietary to the manufacturer of the portable computer using the

1 manufacturer's data structure. That object is synchronized with a corresponding
2 object on the base computer 102. The object on the base computer is accessed
3 using an application program different than the contact management program on
4 the portable computer. The base computer application uses a different data
5 structure to store the various information contained in the object. The two
6 application synchronization modules 106 and 118 operate in combination with
7 portable synchronization manager 104 and desktop synchronization manager 116
8 to ensure that the two objects are properly synchronized and the appropriate data
9 structures are maintained. This may involve translating and/or converting the
10 items or entries in one object to a different format or structure in the corresponding
11 object, such that the corresponding object can be accessed by the appropriate
12 application.

13 The synchronization process is performed independently of the application
14 programs that create and modify the objects being synchronized. The portable
15 synchronization manager 104 and the desktop synchronization manager 116 do not
16 interpret or understand the data entries contained within the synchronized objects.
17 Therefore, the two synchronization managers 104 and 116 merely ensure that the
18 two objects are properly synchronized. Similarly, the applications 108 and 120
19 create and modify objects, but do not participate in the synchronization process.

20 As mentioned above, corresponding objects on the portable computer 100
21 and the base computer 102 typically have different identifiers. The desktop
22 synchronization manager 116 maintains a mapping table of all object identifiers.
23 The mapping table also includes information regarding the volume identifier
24 associated with the objects as well as information regarding whether the object has
25

1 been changed or deleted since the last synchronization process. Table 1 illustrates
2 an exemplary table maintained by desktop synchronization manager 116.

Volume ID	Portable Object ID	Desktop Object ID	Change d	Deleted	Application Information
00	Object1	Object2A	0	0	
01	Object2	Object34	1	0	
02	Object3	Object1F	0	0	
03	Object4	Object27	0	1	

Table 1

11 The Volume ID in Table 1 represents the GUID assigned to a particular storage
12 volume in the portable computer 100. The GUID is represented in Table 1 by a
13 one-byte index rather than using the entire 16 byte GUID. Another table or listing
14 (not shown) is used to identify whether a particular volume is active or inactive
15 (i.e., accessible or in accessible). The corresponding Object IDs are provided in
16 the next two columns of Table 1. The Portable Object ID represents the name of
17 the object used by the portable computer 100 and the Desktop Object ID
18 represents the name of the object used by the desktop computer 102. The Changed
19 and Deleted bits indicate whether an object has been changed or deleted since the
20 last synchronization process. For example, a "0" indicates that the object has not
21 changed or has not been deleted, and a "1" indicates that the object has been
22 modified or deleted since the last synchronization process. The last column in
23 Table 1 is available for storing other information required by particular application
24 programs that have responsibility for the corresponding Volume ID. This
25

1 information can vary from one application program to another. Certain application
2 programs may not require any other information (in which case, the last column of
3 Table 1 is empty. When performing the synchronization process, only objects
4 associated with active volumes are synchronized. All inactive volumes, and the
5 objects stored on those volumes, are ignored during the synchronization process.

6 Fig. 2 shows a general example of a base computer 102 that can be used in
7 accordance with the invention. Computer 102 includes one or more processors or
8 processing units 132, a system memory 134, and a bus 136 that couples various
9 system components including the system memory 134 to processors 132. The bus
10 136 represents one or more of any of several types of bus structures, including a
11 memory bus or memory controller, a peripheral bus, an accelerated graphics port,
12 and a processor or local bus using any of a variety of bus architectures. The
13 system memory 134 includes read only memory (ROM) 138 and random access
14 memory (RAM) 140. A basic input/output system (BIOS) 142, containing the
15 basic routines that help to transfer information between elements within computer
16 102, such as during start-up, is stored in ROM 138.

17 Computer 102 further includes a hard disk drive 144 for reading from and
18 writing to a hard disk (not shown), a magnetic disk drive 146 for reading from and
19 writing to a removable magnetic disk 148, and an optical disk drive 150 for
20 reading from or writing to a removable optical disk 152 such as a CD ROM or
21 other optical media. The hard disk drive 144, magnetic disk drive 146, and optical
22 disk drive 150 are connected to the bus 136 by an SCSI interface 154 or some
23 other appropriate interface. The drives and their associated computer-readable
24 media provide nonvolatile storage of computer-readable instructions, data
25 structures, program modules and other data for computer 102. Although the

1 exemplary environment described herein employs a hard disk, a removable
2 magnetic disk 148 and a removable optical disk 152, it should be appreciated by
3 those skilled in the art that other types of computer-readable media which can
4 store data that is accessible by a computer, such as magnetic cassettes, flash
5 memory cards, digital video disks, random access memories (RAMs), read only
6 memories (ROMs), and the like, may also be used in the exemplary operating
7 environment.

8 A number of program modules may be stored on the hard disk 144,
9 magnetic disk 148, optical disk 152, ROM 138, or RAM 140, including an
10 operating system 158, one or more application programs 160, other program
11 modules 162, and program data 164. A user may enter commands and information
12 into computer 102 through input devices such as a keyboard 166 and a pointing
13 device 168. Other input devices (not shown) may include a microphone, joystick,
14 game pad, satellite dish, scanner, or the like. These and other input devices are
15 connected to the processing unit 132 through an interface 170 that is coupled to
16 the bus 136. A monitor 172 or other type of display device is also connected to the
17 bus 136 via an interface, such as a video adapter 174. In addition to the monitor,
18 personal computers typically include other peripheral output devices (not shown)
19 such as speakers and printers.

20 Computer 102 commonly operates in a networked environment using
21 logical connections to one or more remote computers, such as a remote computer
22 176. The remote computer 176 may be another personal computer, a server, a
23 router, a network PC, a peer device or other common network node, and typically
24 includes many or all of the elements described above relative to computer 102,
25 although only a memory storage device 178 has been illustrated in Fig. 2. The

1 logical connections depicted in Fig. 2 include a local area network (LAN) 180 and
2 a wide area network (WAN) 182. Such networking environments are
3 commonplace in offices, enterprise-wide computer networks, intranets, and the
4 Internet.

5 When used in a LAN networking environment, computer 102 is connected
6 to the local network 180 through a network interface or adapter 184. When used
7 in a WAN networking environment, computer 102 typically includes a modem 186
8 or other means for establishing communications over the wide area network 182,
9 such as the Internet. The modem 186, which may be internal or external, is
10 connected to the bus 136 via a serial port interface 156. In a networked
11 environment, program modules depicted relative to the personal computer 102, or
12 portions thereof, may be stored in the remote memory storage device. It will be
13 appreciated that the network connections shown are exemplary and other means of
14 establishing a communications link between the computers may be used.

15 Generally, the data processors of computer 102 are programmed by means
16 of instructions stored at different times in the various computer-readable storage
17 media of the computer. Programs and operating systems are typically distributed,
18 for example, on floppy disks or CD-ROMs. From there, they are installed or
19 loaded into the secondary memory of a computer. At execution, they are loaded at
20 least partially into the computer's primary electronic memory. The invention
21 described herein includes these and other various types of computer-readable
22 storage media when such media contain instructions or programs for implementing
23 the steps described below in conjunction with a microprocessor or other data
24 processor. The invention also includes the computer itself when programmed
25 according to the methods and techniques described below.

1 For purposes of illustration, programs and other executable program
2 components such as the operating system are illustrated herein as discrete blocks,
3 although it is recognized that such programs and components reside at various
4 times in different storage components of the computer, and are executed by the
5 data processor(s) of the computer.

6 Fig. 3 shows an embodiment of portable computer 100 for use with the
7 present invention. For purposes of this description, the term “portable” is used to
8 indicate a small computing device having a processing unit that is capable of
9 running one or more application programs, a display, and an input mechanism that
10 is typically something other than a full-size keyboard. The input mechanism
11 might be a keypad, a touch-sensitive screen, a track ball, a touch-sensitive pad, a
12 miniaturized QWERTY keyboard, or the like. In other implementations, the
13 portable computer may be implemented as a personal digital assistant (PDA), a
14 personal organizer, a palmtop (or handheld) computer, a computerized notepad, or
15 the like.

16 Portable computer 100 includes an LCD display 200 and several user input
17 keys or buttons 202. The LCD display 200 is a touch-sensitive screen which,
18 when used in conjunction with a stylus 204, allows a user to input information to
19 portable computer 100. The stylus 204 is used to press the display at designated
20 coordinates for user input. Buttons 202 provide another mechanism for user input.
21 A particular portable computer may have any number of buttons for user input.
22 Although not shown in Figure 3, portable computer 100 also includes one or more
23 slots or other connectors capable of receiving removable memory cards.

24 Fig. 4 is a block diagram illustrating pertinent components of the portable
25 computer 100. Portable computer 100 includes built-in memory 110 and one or

1 more removable memory cards 112. Built-in memory 110 includes an operating
2 system 220, one or more application programs 222, and a registry 224.
3 Additionally, portable computer 100 has a processor 228, I/O components 230
4 (including the display 200 and buttons 202 in Fig. 3), and a serial interface 232 for
5 communicating with other computing devices (such as base computer 102 or
6 another portable computer 100). In one embodiment, the various components in
7 portable computer 100 communicate with one another over a bus 234. In an
8 exemplary embodiment of portable computer 100, built-in memory 110 is a non-
9 volatile electronic memory such as a random access memory (RAM) with a
10 battery back-up module, not shown. In an alternate embodiment, built-in memory
11 110 is implemented using a flash memory device. Part of this built-in memory 110
12 is addressable memory for program execution, and the remaining part is used to
13 simulate secondary disk storage.

14 Operating system 220 executes on processor 228 from built-in memory
15 110. In a particular embodiment of the invention, portable computer 100 runs the
16 "Windows CE" operating system manufactured and distributed by Microsoft
17 Corporation of Redmond, Washington. This operating system is particularly
18 designed for small computing devices.

19 Application programs 222 execute from built-in memory 110 of portable
20 computer 100. The number of application programs 222 that can be
21 simultaneously installed on portable computer 100 is a function of the portion of
22 built-in memory allocated to store application programs and the size of the
23 application programs 222 currently installed. In addition, application programs
24 can be installed on removable memory cards 112 as described below.
25

1 The registry 224 is a database that is implemented in various forms under
2 different versions of the "Windows" operating systems. The registry contains
3 information about applications stored on portable computer 100. Exemplary
4 registry information includes user preferences and application configuration
5 information.

6 Fig. 5 is an architectural diagram of a system in accordance with the
7 invention for synchronizing objects between portable computer 100 and base
8 computer 102. As discussed above, desktop synchronization manager 118
9 coordinates the synchronization of objects by determining which objects are stored
10 on storage volumes accessible to the portable computer 100 and synchronizing
11 only those objects that are accessible to the portable computer. Portable
12 synchronization manager 104 operates in combination with desktop
13 synchronization manager 118 to synchronize objects stored on the portable
14 computer with corresponding objects on base computer 102.

15 Communications modules 240 and 242 are implemented on base computer
16 102 and portable computer 100, respectively. These communications modules
17 implement serial communications between the base computer and the portable
18 computer using a serial connection 114 (e.g., a serial cable or an infrared link).
19 Desktop synchronization manager module 118 communicates with various
20 operating system components of portable computer 100 through these
21 communications components.

22 Fig. 6 illustrates multiple volumes currently stored on portable computer
23 100 and base computer 102. Each volume contains one or more objects that are
24 synchronized with corresponding objects in a corresponding volume on the other
25 device. In the example of Fig. 6, the base computer 102 contains eight volumes,

1 labeled Volume 1 through Volume 8. These eight volumes represent all volumes
2 that have been accessible to portable computer 100 during previous
3 synchronization processes and that have not been deleted from the storage devices
4 of the portable computer. Each volume in the desktop computer contains one or
5 more objects. These objects represent a copy of the objects that were stored in the
6 corresponding portable computer volume during the most recent synchronization
7 process (i.e., the most recent synchronization process during which the
8 corresponding portable computer volume was accessible to the portable
9 computer).

10 As shown in Fig. 6, portable computer 100 currently has four accessible
11 volumes (Volumes 1, 4, 5, and 6) and four inaccessible volumes (Volumes 2, 3, 7,
12 and 8). The four inaccessible volumes can be determined by identifying volumes
13 that are currently stored in the base computer (i.e., previously synchronized when
14 the volumes were accessible to the portable computer) but are not currently
15 accessible to the portable computer. The four inaccessible volumes may be
16 removable memory cards that have been removed from the portable computer 100
17 or some other type of storage device that can become temporarily inaccessible to
18 the portable computer (such as an interrupted network connection or a database
19 that is currently off-line). If a synchronization process is initiated with the
20 volumes configured as shown in Fig. 6, objects stored on Volumes 1, 4, 5, and 6 of
21 portable computer 100 will be synchronized with corresponding Volumes 1, 4, 5,
22 and 6 of base computer 102. Objects on any of the other four volumes (Volumes
23 2, 3, 7, or 8) will not be synchronized until the removable memory card containing
24 one or more of the objects is re-inserted into the portable device. Changes made to
25 any objects stored on Volumes 2, 3, 7, or 8 of base computer 102 will be

1 monitored and recorded by the desktop computer. During the next
2 synchronization of each modified object, the recorded changes will be entered.

3 Fig. 7 is a flow diagram illustrating an exemplary procedure for
4 synchronizing objects between a portable computer 100 and a base computer 102.
5 At step 250, the portable synchronization manager 104 identifies volumes that are
6 currently accessible to the portable computer. A function such as "FindObject" is
7 useful to identify all accessible volumes (and the objects stored in those volumes)
8 in the portable computer 100. The FindObjects function is used by the portable
9 synchronization manager 104 to call each application synchronization module 106,
10 which is associated with a particular application 108. Each application
11 synchronization module 106 returns to the portable synchronization manager 104 a
12 list of known volumes (and associated objects) that are currently accessible.

13 The FindObjects function identifies a particular accessible volume in
14 portable computer 100 and identifies all objects associated with the particular
15 volume that need to be synchronized. The objects that need to be synchronized are
16 those that have been modified since previously synchronizing the particular
17 volume. The FindObjects function then returns a list of the identified objects to
18 the portable synchronization manager 106 along with a volume identifier
19 associated with the volume containing the identified objects. The portable
20 synchronization manager 104 then calls the FindObjects function a second time,
21 which allows the application synchronization manager 104 to release resources by
22 deleting the list of objects identified as a result of the first call of FindObjects.
23 The second call of FindObjects causes FindObjects to determine whether
24 additional volumes remain on the portable computer 100 that may contain objects
25 that require synchronization. This second call of the function returns an indication

1 of whether additional volumes remain. If additional volumes remain, then the
2 FindObjects function is called again to retrieve the objects associated with another
3 volume.

4 After all application synchronization modules 106 have responded (using
5 the FindObjects function) with a list of accessible volumes and associated objects,
6 the portable synchronization manager 104 consolidates the multiple lists and
7 communicates the consolidated list of accessible volumes to the desktop
8 synchronization manager 116 (step 252).

9 At step 254, the desktop synchronization manager 116 retrieves a list of
10 volumes previously accessible to the portable computer 100. Once a volume has
11 been identified as accessible to the portable computer, the desktop synchronization
12 manager 116 maintains that volume identifier in the list of previously accessible
13 volumes, regardless of the number of synchronization cycles that have been
14 performed in which the volume was not accessible.

15 Step 256 comprises comparing the list of previously accessible volumes
16 with the list of currently accessible volumes (retrieved in step 254) on the portable
17 computer 100. Each entry in the list of previously accessible volumes indicates
18 the status of the volume at the time of the last synchronization process. The
19 volume status is indicated as either active (i.e., accessible) or inactive (i.e.,
20 inaccessible). Each volume in the list of currently accessible volumes is compared
21 to the corresponding volume in the list of previously accessible volumes to
22 determine whether the status of the volume has changed since the last
23 synchronization process. Also, the list of previously accessible volumes is
24 analyzed to see if any volume having an active status has become inactive (i.e., not
25 on the list of currently accessible volumes).

1 If a volume is currently accessible and was previously accessible, then the
2 volume's status remains active (step 258). If a particular volume is not currently
3 accessible, but was previously accessible, then the volume's status is set to
4 inactive (step 260). If a volume is currently accessible, but was not previously
5 accessible, then the volume is added to the list of previously accessible volumes
6 and the volume's status is set to active (step 262). After comparing each volume
7 and updating the status of the volumes in the list of previously accessible volumes
8 (if necessary), the portable computer 100 and the base computer 102 are
9 synchronized. The synchronization process is initiated by the desktop
10 synchronization manager at step 264. During the synchronization process, objects
11 that have changed since the previous synchronization process and are stored in
12 active volumes are synchronized between the portable computer 100 and the base
13 computer 102. Objects stored on inactive volumes are not synchronized,
14 regardless whether the objects have changed since the previous synchronization.
15 Thus, if an object is stored on a removable memory card that is inactive (i.e.,
16 removed from the portable device), the corresponding object on the base computer
17 is not deleted. Further, the object on the base computer can be modified using its
18 associated application program. The desktop synchronization manager 116
19 continues to monitor and record changes made to objects in inactive volumes (step
20 266), thereby allowing synchronization of the changes with the portable computer
21 100 when the volume again becomes active.

22 Particular embodiments of the invention are described above with reference
23 to a portable computer having one or more removable memory cards. However,
24 the teachings of the present invention can be applied to any computing device
25 capable of accessing a storage device that may become inaccessible. The

1 inaccessibility may be caused by a broken or disabled connection between the
2 storage device and the computing device or insufficient bandwidth to
3 communicate data between the storage device and the computing device.
4 Additionally, some or all of the data on a storage device may be temporarily
5 unavailable or "off-line", thereby causing the data to be inaccessible.

6 Thus, as described above, the invention provides a system and method for
7 selectively synchronizing objects between two devices. The synchronization is
8 performed such that objects stored on inaccessible storage devices are not
9 synchronized. Although the objects stored on inaccessible storage devices are not
10 synchronized, the base computer continues to monitor and record changes made to
11 the corresponding objects stored on the base computer. When the previously
12 inaccessible storage device becomes accessible, a synchronization process is
13 performed such that the two instances of the object are again synchronized. The
14 invention allows the user to continue accessing an object through the base
15 computer, although the storage device on which the object is stored is no longer
16 accessible to the portable computer. Thus, the user of the portable computer can
17 temporarily remove storage cards from the portable computer without concern that
18 objects stored on the removed card will be deleted from the base computer or
19 otherwise made inaccessible by the base computer.

20 Although the invention has been described in language specific to structural
21 features and/or methodological steps, it is to be understood that the invention
22 defined in the appended claims is not necessarily limited to the specific features or
23 steps described. Rather, the specific features and steps are disclosed as preferred
24 forms of implementing the claimed invention.